Analysis of BitTorrent

Mohammad Mannan

School of Computer Science, Carleton University
Analysis of BitTorrent

Outline

► History and motivation
► Features of BitTorrent
► BitTorrent in action
► Security issues
► Applications of BitTorrent
► A few mathematical characteristics
► Concluding remarks
A Little Bit of History

- Project started by Bram Cohen有时候 in May 2001
- KaZaa, Gnutella was getting most of the attention in that time
- “Incentives Build Robustness in BitTorrent” – paper by Bram Cohen. Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, June 2003
- It’s open source — http://sourceforge.net/projects/bittorrent/

a http://bitconjurer.org/
How to Minimize Free-riding?

The biggest social ills that is evident in heavily deployed P2P systems

Difficulties arise from the following P2P criteria:

- large populations
- peers come and go at will
- asymmetry of interest
- collusion
- zero-cost identities
The Tragedy of the Commons

- A well-known theory by Garrett Hardin (1968), Professor of Human Ecology at the University of California, Santa Barbara
- \( N \) herdsmen share a common place for grazing their animals
- Each herdsman seeks to maximize his gain
- “What is the utility to me of adding one more animal to my herd?”
  1. Gets all the benefits from the additional animal (utility is nearly +1)
  2. Overgrazing are shared by all the herdsmen (utility is \(-1/N\))
- The rational herdsman adds one more animal, and so does his fellows — “Ruin is the destination toward which all men rush”
Folklore Version of the Above Phenomenon

Once upon a time, there was a king who wanted to fill a pond with milk from the peasants of his jurisdiction.

He ordered all peasants to bring a small bucket of milk in the morning.

A smart peasant thought as everyone was going to bring milk, if he took a bucket of water, it would not make any difference.

In the morning, the king was astonished to see his pond full of water.
Features of BitTorrent

- The protocol is focused on a single file, instead of a group of files
- Fast replication of a single large popular file
- Simple but strong (in theory and practice) incentive model — serve while you are being served
- Targets of searching are “pieces”, not files
- Requires a centralized element (the tracker)
- Peer selection is evolutionary (starting with a random set of peers)
- Able to sustain flash crowds
Where Does BitTorrent Stand at Present

- CacheLogic surveyed actual P2P traffic from top European ISPs
- They also identified traffic as BitTorrent, KaZaa or Gnutella
- Over six months (Jan to June 2004) of survey data presented

Share of P2P traffic in June 2004:
- BitTorrent: 53%
- eDonkey: 24%
- FastTrack: 19%
- Gnutella: 4%

http://www.cachelogic.com/research/slidel.php
Overview of BitTorrent Protocol

- Publisher creates a .torrent file using BitTorrent software
- Publisher places the .torrent file in a web server (generally)
- A tracker server to coordinate downloading
- To download a file, a peer first connects to the tracker of the file
- The tracker returns a random list of peers that have the file
- The downloader downloads the pieces from others as well as uploads the pieces available to the downloader
A `.torrent File`

- Static file (does not change in the course of protocol run)
- Provides meta-information regarding the file to be shared
  - tracker server’s URL
  - file name
  - file size
  - checksums of pieces
The Tracker

- Helps downloaders to find each other — the place to *rendezvous*
- Does not share any *content*
- Speaks a very simple protocol layered on top of HTTP
- A downloader sends information about: file it is downloaded, port it is listening on etc.
- The tracker responds with a random\(^a\) list of peers which are downloading the same file
- Generally, the web server that publishes the .torrent file, also hosts the tracker for that file

\(^a\)provides robustness
The Protocol in Action – Definitions (1)

- **piece:** A file is broken into pieces of size 256KB each

- **sub-piece:** A piece is broken into sub-pieces of size 16KB each

- **reporting:** Each downloader reports to all of its peers what pieces it has

- **downloader/leecher:** A peer who has zero or more pieces (not all) of a file

- **seeder:** A peer who has all pieces of a file and stays in the torrent network

\[a\] always keeps five sub-piece request pipelined.
The Protocol in Action – Definitions (2)

- **unchoking:** Uploading is called unchoking in BitTorrent. Each peer uploads to four peers who provide the best downloading rate.

- **optimistic unchoking:** Each peer randomly selects a fifth peer to upload and the upload to the peer with the least downloading rate is dropped.

- **choking:** Choking is a temporary refusal to upload. Choking is one peer telling the other s/he needs to contribute more.

---

$^a$ seeders will upload to the peers with best uploading bandwidth.
The Protocol in Action – Piece Selection

- **partial first:** Once a single sub-piece has been requested, the remaining sub-pieces from that piece are requested before sub-pieces from any other piece.

- **rarest first:** Peers download pieces which the fewest of their peers have first.

- **random first piece:** When downloading starts, a peer has nothing to upload. It is important to get a complete piece as quickly as possible. The peer requests a random piece.

- **endgame mode:** Deployed at the end of a download to prevent slow ending. A downloader requests for all sub-pieces of the last piece to all of its downloading peers.

---

*a random piece, when all pieces are equally available*
The Protocol in Action – Choking Algorithm(1)

- **Pareto efficiency**: No two counterparties can make an exchange and both be happier.

- Local optimization algorithm in which pairs of counterparties see if they can improve their lot together, and this may lead to global optima.

- Tit-for-tat/prisoner’s dilemma.

- Selecting which peers to unchoke depends strictly on current download rates\(^a\)

\(^a\)a rolling average of 20 seconds is used to determine download rate.
The Protocol in Action – Choking Algorithm(2)

- Choking decisions are made in every 10 seconds
- Optimistic unchoking is used to explore the network
- Which peer is the optimistic unchoke is rotated every third rechoke period (30 seconds)
- Similar to cooperating on the first move in a prisoner’s dilemma game
Searching for Files

- No options for searching files in the protocol
- Google any specific .torrent file
- There are services designed to facilitate search by accumulating torrent files from well-known torrent hosting websites. E.g.,
  - http://torrentsearch.bounceme.net
  - http://www.btbot.com
  - http://www.watchen.tv
Security Issues

- Anyone with a BitTorrent client can clearly see the IP address of every other user connected to the same tracker.
- File data integrity is ensured via a checksum (SHA-1).
- The tracker distributes all the checksums.
- Upload of data with a failed checksum does not count as uploading — fake uploads help nothing.
- No authentication — nothing to prevent TCP connection hijacking or DNS spoofing.
- DoS attacks on tracker servers are possible.
Applications of BitTorrent

- Lindows is giving customers 50 percent discounts if they download using BitTorrent
- Mozilla distribution tracker hosted by pryan.org
- Windows XP SP2
- Fedora, Slackware Linux distribution
- Blizzard Entertainment using it to distribute the beta of their new game
- “BitTorrent is the future, and it’s the thing that’s going to wreck commercial TV as we know it.”

---

[a] Mark Pesce, Lecturer, Interactive Media, AFTRS (Australian Film Television and Radio School)
Similar Protocols (1)

Slurpie\textsuperscript{a} — InfoComm 2004

\textbf{Pros}

- Less load on the topology server (tracker) and on the primary source (seed) through a back-off algorithm.
- Slurpie is able to outperform BitTorrent in a controlled environment

\textbf{Cons}

- Complex algorithm
- Require to estimate the number of peers in the Slurpie network
- Actual performance of Slurpie in case of flash crowds and for a large number of clients is unknown

\textsuperscript{a} \url{http://www.ieee-infocom.org/2004/Papers/19_3.PDF}
Similar Protocols(2)

- CoopNet (Microsoft Research\textsuperscript{a})
- Intended application – downloading small HTML files
- Flash crowd at web servers
- No notion of serving a partially downloaded file

\textsuperscript{a} http://research.microsoft.com/~padmanab/projects/CoopNet
Effectiveness of File Sharing(1)


Available at:
http://tesla.csl.uiuc.edu/~srikant/Papers/sigcomm04.pdf
Effectiveness of File Sharing (2)

Notation

- $\eta$: effectiveness of file sharing in BitTorrent
- $i$: a given downloader
- $j$: a downloader connected to $i$
- $N$: number of pieces of the served file
- $n_i$: number of pieces of at downloader $i$
- $k$: number of downloaders that $i$ is connected to
- $K$: max number of downloaders that a peer can connect
- $x$: number of downloaders in the system

Here, $k = \min\{x - 1, K\}$
\[ \eta = 1 - \Pr\{\text{downloader } i \text{ has no piece that the connected peers need}\} \\
= 1 - \left(1 - \Pr\{\text{downloader } j \text{ needs no piece from downloader } i\}\right)^k \\
= 1 - \left(1 - \Pr\{\text{downloader } j \text{ has all pieces of downloader } i\}\right)^k \\
\approx 1 - \left(\frac{\log N}{N}\right)^k \]
In BitTorrent, each piece is typically 256KB.

For a modest size file (e.g. 100MB), $N$ is large.

Even if $k = 1$, $\eta$ is very close to 1.

$k$ is actually larger since $K$ is typically 40.
Free Riding and Optimistic Unchoking (1)

$g_1$  a group of peers

$N$  total number of peers in $g_1$

$\mu$  uploading bandwidth of peers in $g_1$ (same for all)

$n_u$  number of uploads of each peer (except the optimistic unchoking upload)

$i$  a given peer in $g_1$

$j$  a peer with zero uploading bandwidth in the network

$\Rightarrow$  Probability that $i$ uploads to $j$ is \( \frac{1}{N-n_u} \)

$\Rightarrow$  Total average downloading rate of peer $j$ is

\[
N \frac{1}{N-n_u} \frac{\mu}{n_u+1} \approx \frac{\mu}{n_u+1}
\]
In BitTorrent, $n_u = 4$

A free-rider gets 20% of the possible maximum downloading rate

Increasing $n_u$ may not be a good idea — more connections, more timeouts, poor performance

Free riders effects are reduced by the seeders (possibly)
Concluding Remarks

- BitTorrent works because it effectively deters free-riders
- New users cannot say they don’t have anything for long
- Downloads may stuck if there is no seeders around
- Seeders are *good* people or they are *lazy*
- Is the current incentive model for seeders good enough?
Thanks.